In response to the objection to the specification, the description has been amended above.

Regarding the rejections of the claims, applicant respectfully traverses the assertions in the previous office actions.

In response to the rejection of claims 1-3 and 5-7 under 35 U.S.C. §102(b), for allegedly being anticipated by WO 98/09290 to David, the citation does not identically disclose all the elements of the claimed invention.

More specifically with regard to claims 1 and 7, the citation does not suggest "instruction for a conditional arithmetical operation," as in claims 1 and 7. The only conditional instruction disclosed in David is a conditional offset to a list. An instruction for performing a conditional offset to a list does not suggest an instruction for performing a conditional arithmetic operation.

With regard to claim 5, the citation does not suggest "instructions for conditional arithmetical operations," as in claim 5.

The claims are definite and distinguished from the citations and Applicant respectfully requests the allowance of all claims.

The Commissioner is hereby authorized to credit any overpayment or charge any fee (except the issue fee) including fees for any required extension of time, to Account No. 14-127[illegible].

Respectfully submitted,

By Michael Belk
Michael E. Belk, Reg. 33,35[illegible]
Patent Attorney
(914) 333-9643

CERTIFICATE OF TRANSMISSION
I hereby certify that this correspondence is being facsimile transmitted to the Patent and Trademark Office at Fax number: 703-872-9314
On January 4, 2002
(Transmission Date)

By Michael E Belk
(Signature)
S:\BE\PN04BEA0.BER.DOC

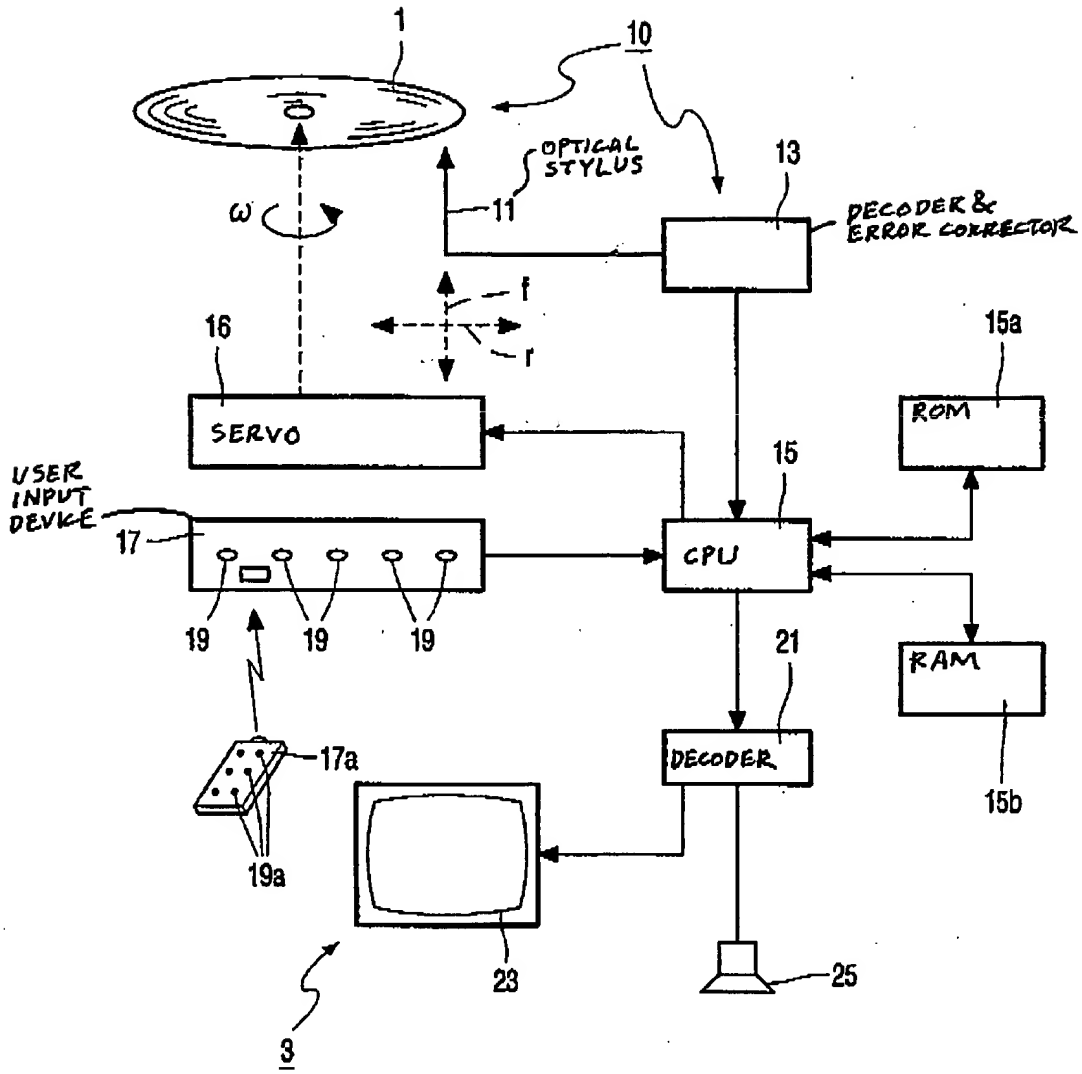
1
10
OPTICAL STYLUS
11
13
DECODER & ERROR CORRECTOR
ω
f
r
16
SERVO
15a
ROM
USER INPUT DEVICE
17
15
CPU
19
19
19
21
RAM
DECODER
17a
19a
15b
23
25
3

FIG. 1

Field name	Size (Bits)
Command List Header	8
Instruction	40
Next_List Offset	16

FIG. 4

Byte #1 0..2	Byte #1 3..4	Byte #1 5..7	Byte #2	Byte #3	Byte #4	Byte #5	Abbreviation	Description
001	cond0	calc	i	j	k	l	calc	if cond0 { V[j] := V[k] opcode V[ ] }
010	cond0	000	i	--	k	l	move0	if cond0 { V[k] := V[ ] }
	cond1	001	i	j	k	l	move1	if cond1 { V[k] := V[ ] }
	cond0	010	i	j	dd	dd	fill0	if cond0 { V[j] := dddd }
	00	110	i	j	dd	dd	fillr	while V[i] > = 0 { V[j + V[i]] := dddd; V[i]-- }
100	cond0	000	i	--	offs	offs	jump0	if cond0 { goto offs }
	cond1	001	i	j	offs	offs	jump1	if cond1 { goto offs }
	cond0	010	i	j	--	--	return	if cond0 { goto V[j] }
	cond0	100	i	--	offs	offs	loop0	if cond0 { dec (V[i]; goto offs }
	cond1	101	i	j	offs	offs	loop1	if cond1 { dec (V[i]; goto offs }
	cond0	110	i	j	offs	offs	loop2	if cond0 { dec (V[i], V[j]; goto offs }
110	cond0	000	i	j	offs	offs	jumpw	if cond0 { wait V[j] seconds for input; goto offs }

FIG. 5